# Git User Guide

A reference for git workflow

Written by Christian Heimvik

## Normal workflow for first time

1. Clone the repository:

   ```
   git clone <repository-url>
   ```

2. Navigate to the repository:

   ```
   cd <repository-name>
   ```

3. Do your work in the repository...

4. Add changes to staging area:

   ```
   git add .
   ```

5. Commit the changes:

   ```
   git commit -m "Your␣commit␣message"
   ```

6. Push the changes to remote repository:

   ```
   git push origin <branch-name>
   ```

## Workflow for subsequent work

1. Pull the latest changes from the remote repository:

   ```
   git pull origin <branch-name>
   ```

2. Do your work in the repository...

3. Add changes to staging area:

   ```
   git add .
   ```

4. Commit the changes:

   ```
   git commit -m "Your␣commit␣message"
   ```

5. Push the changes to remote repository:

   ```
   git push origin <branch-name>
   ```

## Handling merge conflicts

1. Resolve the conflicts manually in the affected files.

2. Add the resolved files to staging:

   ```
   git add <file-name>
   ```

3. Commit the resolved changes:

   ```
   git commit -m "Resolved␣merge␣conflict"
   ```

4. Continue with the push:

   ```
   git push origin <branch-name>
   ```

## Branching and checking out other branches

1. Create a new branch:

   ```
   git branch <new–branch–name>
   ```

2. Switch to the new branch:

   ```
   git checkout <new–branch–name>
   ```

   Alternatively, create and switch to a new branch in one step:

   ```
   git checkout –b <new–branch–name>
   ```

3. Push the new branch to the remote repository:

   ```
   git push origin <new–branch–name>
   ```

4. List all branches (local and remote):

   ```
   git branch –a
   ```

5. Delete a branch locally after it has been merged:

   ```
   git branch –d <branch–name>
   ```

6. Delete a branch remotely (after merging):

   ```
   git push origin —delete <branch–name>
   ```

## Stashing changes

If you need to save your work temporarily without committing:

1. Stash your current changes:

   ```
   git stash
   ```

2. List all stashed changes:

   ```
   git stash list
   ```

3. Apply the most recent stash:

   ```
   git stash apply
   ```

4. Apply a specific stash by index:

   ```
   git stash apply stash@{<index>}
   ```

5. Drop a specific stash after applying it:

   ```
   git stash drop stash@{<index>}
   ```

6. Clear all stashes:

   ```
   git stash clear
   ```

## Global authentication

If you encounter an author identity error, set your global username and email:

```
git config —global user.name ”Your␣Name”
git config —global user.email ”your_email@example.com”
```